

CERTIFICATE OF MAILING BY "EXPRESS MAIL" UNDER 37 CFR § 1.10

"Express Mail" mailing label number: EH 861 966 417 US

Date of Mailing: 7/31/98

I hereby certify that the documents indicated below are being deposited with the United States Postal Service under 37 CFR 1.10 on the date indicated above and are addressed to Box Patent Applications, Commissioner of Patents and Trademarks, Washington, D. C. 20231 and mailed on the above Date of Mailing with the above "Express Mail" mailing label number.

Martha A. Acosta *Martha A. Acosta*

Typed or printed name of person mailing paper of fee: SIGNATURE of person mailing paper of fee

A

07/31/98  
Jc500 U.S. PTO

ASSISTANT COMMISSIONER OF PATENTS  
Washington, D C. 20231

DOCKET NUMBER: AT9-98-302  
July 31, 1998

Jc398 U.S. PTO  
09/127336  
07/31/98

Sir:  
Transmitted herewith for filing is the Patent Application of:

Inventor: Bruce A. Beadle et al..

For: METHOD AND APPARATUS FOR SELECTIVELY AND DYNAMICALLY SELECTING CLASSES IN A DATA PROCESSING SYSTEM

Enclosed are:

- ☒ Patent Specification and Declaration
- ☒ 9 sheets of drawing(s).
- ☒ An assignment of the invention to International Business Machines Corporation (includes Recordation Form Cover Sheet).
- ☐ A certified copy of a application.
- ☐ Information Disclosure Statement, PTO 1449 and copies of references.

The filing fee has been calculated as shown below:

For	Number Filed	Number Extra	Rate	Fee
Basic Fee				\$790
Total Claims	19 - 20	-0-	x 22 =	\$
Indep. Claims	4 - 3	-1-	x 82 =	\$ 82
MULTIPLE DEPENDENT CLAIM PRESENTED			x 270 =	\$
TOTAL				\$ 872.

☒ Please charge my Deposit Account No. 09-0447 in the amount of \$ 872 . A duplicate copy of this sheet is enclosed.

☒ The Commissioner is hereby authorized to charge payment of the following fees associated with this communication or credit any overpayment to Deposit Account 09-0447. A duplicate copy of this sheet is enclosed.

☒ Any additional filing fees required under 37 CFR §1.16.

☒ Any patent application processing fees under 37 CFR §1.17.

Respectfully submitted,

By *Richard A. Henkler*  
Richard A. Henkler  
Registration No. 39,220  
Intellectual Property Law Dept.  
IBM Corporation  
11400 Burnet Road - 4054  
Austin, Texas 78758  
Telephone (512) 823-0962

09/127336

Docket No. AT9-98-302

**METHOD AND APPARATUS FOR SELECTIVELY AND DYNAMICALLY  
SELECTING CLASSES IN A DATA PROCESSING SYSTEM**

5

**BACKGROUND OF THE INVENTION**

**CROSS-REFERENCE TO RELATED APPLICATION**

The present invention is related to the following applications entitled "Method and Apparatus to  
10 Selectively Define Java Virtual Machine Initializing Properties Using a Browser Graphical User Interface", Attorney Docket AT9-98-305, filed even date hereof, assigned to a common assignee, and "Method and Apparatus for Selecting a Java Virtual Machine For Use With a  
15 Browser", Attorney Docket No. AT9-98-306, filed even date hereof, are incorporated herein by reference.

**1. Technical Field:**

The present invention provides an improved data  
20 processing system and in particular an improved method and apparatus for selecting properties for a JVM. Still more particularly, the present invention provides a method and apparatus for selecting classes for a JVM used with a browser.

25

**2. Description of Related Art:**

Internet, also referred to as an "internetwork", in communications is a set of computer networks, possibly dissimilar, joined together by means of gateways that  
30 handle data transfer and the conversion of messages from

Docket No. AT9-98-302

the sending network to the protocols used by the receiving network (with packets if necessary). When capitalized, the term "Internet" refers to the collection of networks and gateways that use the TCP/IP suite of protocols.

5           The Internet has become a cultural fixture as a source of both information and entertainment. Many businesses are creating Internet sites as an integral part of their marketing efforts, informing consumers of the products or services offered by the business or providing  
10 other information seeking to engender brand loyalty. Many federal, state, and local government agencies are also employing Internet sites for informational purposes, particularly agencies which must interact with virtually all segments of society such as the Internal Revenue  
15 Service and secretaries of state. Operating costs may be reduced by providing informational guides and/or searchable databases of public records online.

          Currently, the most commonly employed method of transferring data over the Internet is to employ the  
20 World Wide Web environment, also called simply "the web". Other Internet resources exist for transferring information, such as File Transfer Protocol (FTP) and Gopher, but have not achieved the popularity of the web. In the web environment, servers and clients effect data  
25 transaction using the Hypertext Transfer Protocol (HTTP), a known protocol for handling the transfer of various data files (e.g., text, still graphic images, audio, motion video, etc.). Information is formatted for presentation to a user by a standard page description  
30 language, the Hypertext Markup Language (HTML). In

2025 RELEASE UNDER E.O. 14176

Docket No. AT9-98-302

addition to basic presentation formatting, HTML allows developers to specify "links" to other web resources identified by a Uniform Resource Locator (URL). A URL is a special syntax identifier defining a communications path to specific information. Each logical block of information accessible to a client, called a "page" or a "web page", is identified by a URL. The URL provides a universal, consistent method for finding and accessing this information by the web "browser". A browser is a program capable of submitting a request for information identified by a URL at the client machine. Retrieval of information on the web is generally accomplished with an HTML-compatible browser.

When a user desires to retrieve a page, a request is submitted to a server connected to a client computer at which the user is located and may be handled by a series of servers to effect retrieval of the requested information. The information is provided to the client formatted according to HTML. Typically, personal computers (PCs) along with work stations are typically used to access the Internet.

Often applications or programs may be sent to a computer from a web server across the Internet. Java applications are becoming increasingly more prevalent as the type of application sent between web servers and client computers. Java applications are common on the Internet and becoming more increasingly common in intranets and in other types of networks used in businesses.

Docket No. AT9-98-302

Java is an object oriented programming language and environment focusing on defining data as objects and the methods that may be applied to those objects. Java supports only a single inheritance, meaning that each  
5 class can inherit from only one other class at any given time. Java also allows for the creation of totally abstract classes known as interfaces, which allow the defining of methods that may be shared with several classes without regard for how other classes are handling  
10 the methods.

The Java virtual machine (JVM) is a virtual computer component that resides only in memory. The JVM allows Java programs to be executed on a different platform as opposed to only the one platform for which the code was  
15 compiled. Java programs are compiled for the JVM. In this manner, Java is able to support applications for many types of data processing systems, which may contain a variety of central processing units and operating systems architectures. To enable a Java application to  
20 execute on different types of data processing systems, a compiler typically generates an architecture-neutral file format - the compiled code is executable on many processors, given the presence of the Java run-time system. The Java compiler generates bytecode  
25 instructions that are non-specific to a particular computer architecture. A bytecode is a machine independent code generated by the Java compiler and executed by a Java interpreter. A Java interpreter is a part in the JVM that alternately decodes and interprets a  
30 bytecode or bytecodes. These bytecode instructions are

Docket No. AT9-98-302

designed to be easy to interpret on any computer and easily translated on the fly into native machine code.

Presently available browsers are designed with a notion of a fixed JVM, which uses a fixed value for the Java classpath. No flexible mechanisms are presently available for modifying or viewing the classpath within a browser. The problem with this approach is that of the global nature of the changing system classpath requires the user to modify the system defined global classpath variable manually or via a script which is executed prior to executing the browser. This system classpath variable is shared with other non-Java browser related applets or applications in addition to the Java enabled browser. Another problem exists for browsers that provide for a multi-user environment in which multiple users/user profiles are being employed. In such a situation, each user profile is forced to use the same environment.

Therefore, it would be advantageous to have an improved method and apparatus for providing users an ability to use more recent versions of JVMs without having to wait for an updated version of the web browser.

Docket No. AT9-98-302

### SUMMARY OF THE INVENTION

The present invention provides a method for  
5 selecting classes using a browser for use by a virtual  
machine in a data processing system. The browser  
provides an interface in which the interface allows for  
selection of classes for use by the virtual machine. A  
selection of classes is received through the interface.  
10 The selection of classes is stored by the browser,  
wherein the selection of classes is used by the browser  
when initializing the virtual machine.

15

2010 RELEASE UNDER E.O. 14176

Docket No. AT9-98-302

### BRIEF DESCRIPTION OF THE DRAWINGS

5           The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed  
10 description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

**Figure 1** is a pictorial representation of a distributed data processing system in which the present invention may be implemented;

15           **Figure 2** is a block diagram of a data processing system which may be implemented as a server in accordance to the present invention;

**Figure 3** is a block diagram of a data processing system in which the present invention may be implemented;

20           **Figure 4** is a diagram of components used to select classpath value in accordance with a preferred embodiment of the present invention;

**Figure 5** is a graphical user interface used in modifying classpath values in accordance with a preferred  
25 embodiment of the present invention;

**Figure 6** is a diagram of a user profile data structure managed by a user profile manager in accordance with a preferred embodiment of the present invention;



**Figure 7** is a high level flowchart of a process used to select a JVM for use with a web browser in accordance with a preferred embodiment of the present invention;

5

**Figure 9** is a diagram of a JVM initialization data structure in accordance with a preferred embodiment of the present invention.

Docket No. AT9-98-302

#### DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference now to the figures, and in particular  
5 with reference to **Figure 1**, a pictorial representation of  
a distributed data processing system in which the present  
invention may be implemented is depicted.

Distributed data processing system **100** is a network  
of computers in which the present invention may be  
10 implemented. Distributed data processing system **100**  
contains a network **102**, which is the medium used to  
provide communications links between various devices and  
computers connected together within distributed data  
processing system **100**. Network **102** may include permanent  
15 connections, such as wire or fiber optic cables, or  
temporary connections made through telephone connections.

In the depicted example, a server **104** is connected to  
network **102** along with storage unit **106**. In addition,  
clients **108**, **110**, and **112** also are connected to a network  
20 **102**. These clients **108**, **110**, and **112** may be, for example,  
personal computers or network computers. For purposes of  
this application, a network computer is any computer,  
coupled to a network, which receives a program or other  
application from another computer coupled to the network.  
25 In the depicted example, server **104** provides data, such as  
boot files, operating system images, and applications to  
clients **108-112**. Clients **108**, **110**, and **112** are clients to  
server **104**. Distributed data processing system **100** may  
include additional servers, clients, and other devices not  
30 shown. In the depicted example, distributed data

Docket No. AT9-98-302

processing system **100** is the Internet with network **102** representing a worldwide collection of networks and gateways that use the TCP/IP suite of protocols to communicate with one another. At the heart of the

5 Internet is a backbone of high-speed data communication lines between major nodes or host computers, consisting of thousands of commercial, government, educational, and other computer systems, that route data and messages. Of course, distributed data processing system **100** also may be

10 implemented as an umber of different types of networks, such as for example, an intranet or a local area network.

**Figure 1** is intended as an example, and not as an architectural limitation for the processes of the present invention.

15 Referring to **Figure 2**, a block diagram of a data processing system which may be implemented as a server, such as server **104** in **Figure 1**, is depicted in accordance to the present invention. Data processing system **200** may be a symmetric multiprocessor (SMP) system including a

20 plurality of processors **202** and **204** connected to system bus **206**. Alternatively, a single processor system may be employed. Also connected to system bus **206** is memory controller/cache **208**, which provides an interface to local memory **209**. I/O bus bridge **210** is connected to system bus

25 **206** and provides an interface to I/O bus **212**. Memory controller/cache **208** and I/O bus bridge **210** may be integrated as depicted.

Peripheral component interconnect (PCI) bus bridge **214** connected to I/O bus **212** provides an interface to PCI

30 local bus **216**. A number of modems **218-220** may be

Docket No. AT9-98-302

connected to PCI bus **216**. Typical PCI bus implementations will support four PCI expansion slots or add-in connectors. Communications links to network computers **108-112** in **Figure 1** may be provided through modem **218** and  
5 network adapter **220** connected to PCI local bus **216** through add-in boards.

Additional PCI bus bridges **222** and **224** provide interfaces for additional PCI buses **226** and **228**, from which additional modems or network adapters may be  
10 supported. In this manner, server **200** allows connections to multiple network computers. A memory mapped graphics adapter **230** and hard disk **232** may also be connected to I/O bus **212** as depicted, either directly or indirectly.

Those of ordinary skill in the art will appreciate  
15 that the hardware depicted in **Figure 2** may vary. For example, other peripheral devices, such as optical disk drive and the like also may be used in addition or in place of the hardware depicted. The depicted example is not meant to imply architectural limitations with respect  
20 to the present invention.

The data processing system depicted in **Figure 2** may be, for example, an IBM RISC/System 6000 system, a product of International Business Machines Corporation in Armonk, New York, running the Advanced Interactive Executive (AIX)  
25 operating system.

With reference now to **Figure 3**, a block diagram of a data processing system in which the present invention may be implemented is illustrated. Data processing system **300** is an example of a client computer. Data processing  
30 system **300** employs a peripheral component interconnect

Docket No. AT9-98-302

(PCI) local bus architecture. Although the depicted example employs a PCI bus, other bus architectures such as Micro Channel and ISA may be used. Processor 302 and main memory 304 are connected to PCI local bus 306 through PCI bridge 308. PCI bridge 308 also may include an integrated memory controller and cache memory for processor 302. Additional connections to PCI local bus 306 may be made through direct component interconnection or through add-in boards. In the depicted example, local area network (LAN) adapter 310, SCSI host bus adapter 312, and expansion bus interface 314 are connected to PCI local bus 306 by direct component connection. In contrast, audio adapter 316, graphics adapter 318, and audio/video adapter (A/V) 319 are connected to PCI local bus 306 by add-in boards inserted into expansion slots. Expansion bus interface 314 provides a connection for a keyboard and mouse adapter 320, modem 322, and additional memory 324. SCSI host bus adapter 112 provides a connection for hard disk drive 326, tape drive 328, and CD-ROM 330 in the depicted example. Typical PCI local bus implementations will support three or four PCI expansion slots or add-in connectors.

An operating system runs on processor 302 and is used to coordinate and provide control of various components within data processing system 300 in **Figure 1**. The operating system may be a commercially available operating system such as OS/2, which is available from International Business Machines Corporation. "OS/2" is a trademark of from International Business Machines Corporation. An object oriented programming system such as Java may run in conjunction with the operating system and provides calls

Docket No. AT9-98-302

to the operating system from Java programs or applications  
executing on data processing system 300. Instructions for  
the operating system, the object-oriented operating  
system, and applications or programs are located on  
5 storage devices, such as hard disk drive 326 and may be  
loaded into main memory 304 for execution by processor  
302.

Those of ordinary skill in the art will appreciate  
that the hardware in **Figure 3** may vary depending on the  
10 implementation. For example, other peripheral devices,  
such as optical disk drives and the like may be used in  
addition to or in place of the hardware depicted in **Figure**  
**3**. The depicted example is not meant to imply  
architectural limitations with respect to the present  
15 invention. For example, the processes of the present  
invention may be applied to multiprocessor data processing  
system.

The present invention provides a method, apparatus,  
and instructions for selectively and dynamically loading  
20 classes. This mechanism also enables and enhances  
polymorphism in an otherwise static Java classpath name  
space. The present invention provides a mechanism for  
modifying or viewing the value of a class path for a JVM  
that is used with a browser. Currently available  
25 mechanisms for modifying the classpath variable for  
loading needed classes rely on the user modifying the  
system defined global class path manually or via a script  
executed prior to running the browser. The classpath  
variable is an environmental variable that defines a path  
30 to the "classes.zip" file, which is used to load classes

Docket No. AT9-98-302

for use by the JVM. The present invention provides a high level of integration, which provides users with maximum flexibility in setting the value for a classpath for individual and multiple user environments. Different user profiles may employ different classpaths such that switching of classpaths may be accomplished by selection of a user profile. A graphical user interface (GUI) is provided as part of the browser to allow a user to specify a classpath, also referred to as an "extended classpath". This extended classpath when set within the GUI provides the ability to assign a browser a localized instance per user profile of a classpath. In the depicted examples, a user may choose to append the extended classpath to the beginning or end of the system defined classpath or simply ignore the system defined classpath and only pass the extended classpath to JVM. By changing or appending classpaths, the classes loaded and used in the JVM may be changed. As used herein, the term "browser" refers to browsers in addition to hypertext markup language (HTML) browsers, such as Netscape Communicator for OS/2. In addition, a browser may encompass other applications that operate in a Java based network or other distributed network.

In the depicted examples, the processes and interfaces described are for a browser, such as Netscape Communicator operating in an OS/2 operating system. These examples are not intended to limit the invention to a particular browser or operating system. The processes and interfaces of the present invention may be applied to other types of browsers and operating systems.

Docket No. AT9-98-302

With reference now to **Figure 4**, a diagram of components used to select classpath value is depicted in accordance with a preferred embodiment of the present invention. Internet browser **400** contains a user profile manager **402**, which is employed to manage one or more user profiles for internet browser **400**. Internet browser **400** may be implemented using a browser, such as, for example, Netscape Communicator, which is available from Netscape Communications Corporation. Selection module **404** contains the processes used in providing a user an ability to select classpath values. Selection module **404** presents a GUI **406** to the user, which allows the user to select a value for the classpath that is to be employed for the user when a JVM is initialized in association with the browser. Selection module **404** queries user profile manager **402** within internet browser **400** for user profile information to display to a user in setting a classpath value. Selection module **404** queries application environment **408** for the system classpath. The application environment in the depicted examples is the value of environmental variables before starting the browser. In response, application environment **408** returns the system classpath to selection module **404**. This information is displayed to the user through the GUI **406**. Selections or changes in JVMs are received as user input through GUI **406**. This user input is returned to user profile manager **402** when a JVM is to be started by internet browser **400**, the user profile information with the classpath value is employed to start JVM **410**.



Docket No. AT9-98-302

With reference next to **Figure 5**, a graphical user interface used in modifying classpath values is depicted in accordance with a preferred embodiment of the present invention. Java advanced properties dialog **500** is

5 displayed to a user to allow for modification of various options for using a JVM with a browser. Optional classpath field **502** is present in Java advanced properties dialog **500** to allow a user to enter a classpath. The user then may select a number of options **504**, **506**, and **508**.

10 Option **504** is selected if the user desires to ignore changes to the classpath. This option is the one selected in the depicted example. Option **506** is selected if a user desires to append the classpath to the beginning of the system classpath. Option **508** is selected if a user desires

15 to append the classpath to the end of the system classpath. The current classpath is displayed to the user in current classpath field **510** in Java advanced properties dialog **500**. By selecting option **512**, a user may choose to ignore the system classpath and use only the classpath

20 entered by the user. If the user is satisfied with the changes within Java advanced properties dialog **500**, the user may select OK button **514**. Cancel button **516** is selected if the user does not want to use options changed in Java advanced properties dialog **500**.

25 With reference now to **Figure 6**, a diagram of a user profile data structure managed by a user profile manager is depicted in accordance with a preferred embodiment of the present invention. User profile data structure **600** contains information used to configure behavior of the web

30 browser for a particular user. In the depicted example,

Docket No. AT9-98-302

user profile data structure 600 includes a profile name 602, a Java class path 604, Java parameters 606, a Java path 608, and a Java class path option 610. Profile name 602 is used to uniquely identify the profile from other profiles when the browser contains multiple user profiles. Java class path 604 is used to identify the path in which classes are loaded for use by the JVM. Java parameters 606 contain parameters used by a JVM when the browser initializes or starts a JVM for use with the browser.

These parameters may include, for example, initial heap size, garbage collection information, Java stack size, and reporting options for JVM information. Java path 608 includes the path and file name for the JVM that is to be used with the browser. Java class path option 610 provides information that may be used to depend an extended class path to the beginning or end of the system defined class path. User profile data structure 600 also includes other information (not shown) employed to define the behavior of the browser.

With reference now to **Figure 7**, a high level flowchart of a process used to select a JVM for use with a web browser is depicted in accordance with a preferred embodiment of the present invention. The process begins by obtaining user profile data (step 700). This data is obtained from the user profile manager within the browser. Next, the system classpath is obtained from the application environment (step 702). The data is displayed to a user using a GUI (step 704). A determination is then made as to whether the classpath is to be changed (step 706). For example, this step may be used to change the

Docket No. AT9-98-302

value for the Java classpath that is used to load classes for use by the JVM, which is used with the browser. More specifically in the depicted example, an extended classpath may be appended to the beginning or end of the system classpath, or only the extended classpath may be  
5 passed to the JVM.

If the classpath is not to be changed, the process terminates. Otherwise, the new classpath is obtained (step 708). The new classpath is obtained through a GUI  
10 presented to the user in which the user may select the classpath for use by the JVM to be used with the browser. Thereafter, the new settings for the classpath are written or sent to the profile manager in the browser (step 710) with the process terminating thereafter. The classpath  
15 may be, for example, OS2.ibm.java.classpath.

Turning next to **Figure 8**, a flowchart of a process for starting a JVM using classpath values is depicted in accordance with a preferred embodiment of the present invention. The process begins by checking the profile  
20 manager to determine whether the classpath "OS2.ibm.java.classpath" exists in the user profile data structure for a user profile (step 800). This classpath is the user defined classpath. Next, a determination is made as to whether the classpath is set in the profile manager (step 802). This determination is made by  
25 checking the user profile data structure for the selected user. If the user classpath is set in the profile manager, the process then initializes the JVM initialization arguments (Initargs) data structure to use  
30 the classpath in the user profile (step 804). Then, an

Docket No. AT9-98-302

instance of the virtual machine is created (step 806) with the process terminating thereafter.

With reference again to step 802, if the classpath is not set in the profile manager, a determination is then made as to whether the user wants to use the system class path (step 808). If the user does not want to use the system classpath, the process then proceeds to step 806 to create an instance of the JVM. The instance of the JVM is created by calling a Java native interface (JNI) application programming interface (API). A Java native interface is a native programming interface that allows Java code that runs inside a Java virtual machine to interoperate with applications and libraries written in other programming languages, such as C, C++ and assembly. In the depicted example, the JNI API provides an interface for native applications, such as a browser, to reach Java. The JNI is used to translate messages from other Java objects or components into calls used by the browser and to translate responses from the browser into messages that are recognized by Java objects or components. An example of a JNI API is JNI\_CreateJavaVM. JNI are found in the Java Development Kit (JDK) available from Sun Microsystems, Inc. If the user does want to use the system classpath, the JVM is initialized from InitOrgs data structure to use the system classpath name (step 810) with the process terminating thereafter. Depending on the implementation, the user may actually be an application that uses the processes of the present invention to select or change classpaths. In such an implementation, the GUI interfaces may be bypassed with the selections being made

Docket No. AT9-98-302

directly by the application to the user profile data structure.

Turning next to **Figure 9**, a diagram of a JVM initialization data structure is depicted in accordance with a preferred embodiment of the present invention. JVM initialization data structure **900** illustrates various parameters that are used as a JNI API to initialize a JVM.

Thus, the present invention provides an improved method, apparatus, and instructions for allowing flexibility in selecting classpaths used to load classes for a JVM. For example, the present invention is useful in placing improved classes ahead of standard classes having the same name such that these improved classes are given preference in loading for use with the JVM. In particular, the present invention provides a GUI that allows a user to specify classpaths for use with a JVM when the JVM is initiated by the browser. When switching to a different JVM, the classpaths also may be switched to optimize performance of the JVM.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of the present invention are capable of being distributed in a form of a computer readable medium of instructions and a variety of forms and that the present invention applies equally regardless of the particular type of signal bearing media actually used to carry out the distribution. Examples of computer readable media

Docket No. AT9-98-302

include recordable-type media such a floppy disc, a hard disk drive, a RAM, and CD-ROMs and transmission-type media such as digital and analog communications links.

The description of the present invention has been  
5 presented for purposes of illustration and description,  
but is not limited to be exhaustive or limited to the  
invention in the form disclosed. Many modifications and  
variations will be apparent to those of ordinary skill in  
the art. For example, the processes of the present  
10 invention may be applied to change classpaths dynamically  
or at run time. The embodiment was chosen and described  
in order to best explain the principles of the invention,  
the practical application, and to enable others of  
ordinary skill in the art to understand the invention for  
15 various embodiments with various modifications as are  
suited to the particular use contemplated.

2025 RELEASE UNDER E.O. 14176

Docket No. AT9-98-302

1   **CLAIMS:**

2   What is claimed is:

1   1.   A method for selecting classes using a browser for  
2   use by a virtual machine in a data processing system, the  
3   method comprising:  
4       providing through the browser, an interface in which  
5   the interface allows for selection of classes for use by  
6   the virtual machine;  
7       receiving a selection of classes through the  
8   interface; and  
9       storing the selection of classes, wherein the  
10   selection of classes is used by the browser when  
11   initializing the virtual machine.

1   2.   The method of claim 1, wherein the interface is a  
2   graphical user interface.

1   3.   The method of claim 1, wherein the virtual machine  
2   is a Java virtual machine.

1   4.   The method of claim 1 further comprising:  
2       using the selection of the classes to initialize the  
3   virtual machine.

1   5.   The method of claim 1, wherein the step of storing  
2   the selection of classes comprises storing the selection  
3   of classes in a user profile.

Docket No. AT9-98-302

1 6. The method of claim 1, wherein the selection of  
2 classes is a class path.

1 7. The method of claim 1, wherein the selection of  
2 classes causes a class path to be appended a system class  
3 path.

1 8. A method for selecting classes for use by a Java  
2 virtual machine associated with a browser, the method  
3 comprising:

4 displaying a graphical user interface in which a  
5 classpath may be selected to define classes for use with  
6 the Java virtual machine;

7 receiving a selection of the classpath from the  
8 graphical user interface;

9 storing the selection of the classpath; and

10 initializing the Java virtual machine using the  
11 selection of the class path.

1 9. The method of claim 8, wherein the selection of the  
2 classpath appends the classpath to a system classpath.

1 10. The method of claim 9, wherein the system class path  
2 has an end and wherein the classpath is appended to the  
3 end of the system classpath.

1 11. The method of claim 9, wherein the system classpath  
2 has a beginning and wherein the classpath is appended to  
3 a beginning of the system classpath.



1 12. A data processing system for selecting classes using  
2 a browser for use by a virtual machine in the data  
3 processing system, the data processing system comprising:  
4 providing means for providing through the browser an  
5 interface in which the interface allows for selection of  
6 classes for use by the virtual machine;  
7 receiving means for receiving a selection of classes  
8 through the interface; and  
9 storing means for storing the selection of classes,  
10 wherein the selection of classes is used by the browser  
11 when initializing the virtual machine.

1 14. The data processing system of claim 12, wherein the  
2 virtual machine is a Java virtual machine.

1 16. The data processing system of claim 12, wherein the  
2 step of storing the selection of classes comprises  
3 storing the selection of classes in a user profile.

1 17. The data processing system of claim 12, wherein the  
2 selection of classes is a classpath.

Docket No. AT9-98-302

1 18. The data processing system of claim 12, wherein the  
2 selection of classes causes a classpath to be appended a  
3 system class path.

1 19. A computer program product in a computer readable  
2 medium for selecting classes for use by a Java virtual  
3 machine associated with a browser, the computer program  
4 product comprising:

5 first instructions for displaying a graphical user  
6 interface in which a classpath may be selected to define  
7 classes for use with the Java virtual machine;

8 second instructions for receiving a selection of the  
9 classpath from the graphical user interface;

10 third instructions for storing the selection of the  
11 classpath; and

12 fourth instructions for initializing the Java  
13 virtual machine using the selection of the classpath.

2025 RELEASE UNDER E.O. 14176

Docket No. AT9-98-302

**ABSTRACT OF THE DISCLOSURE**

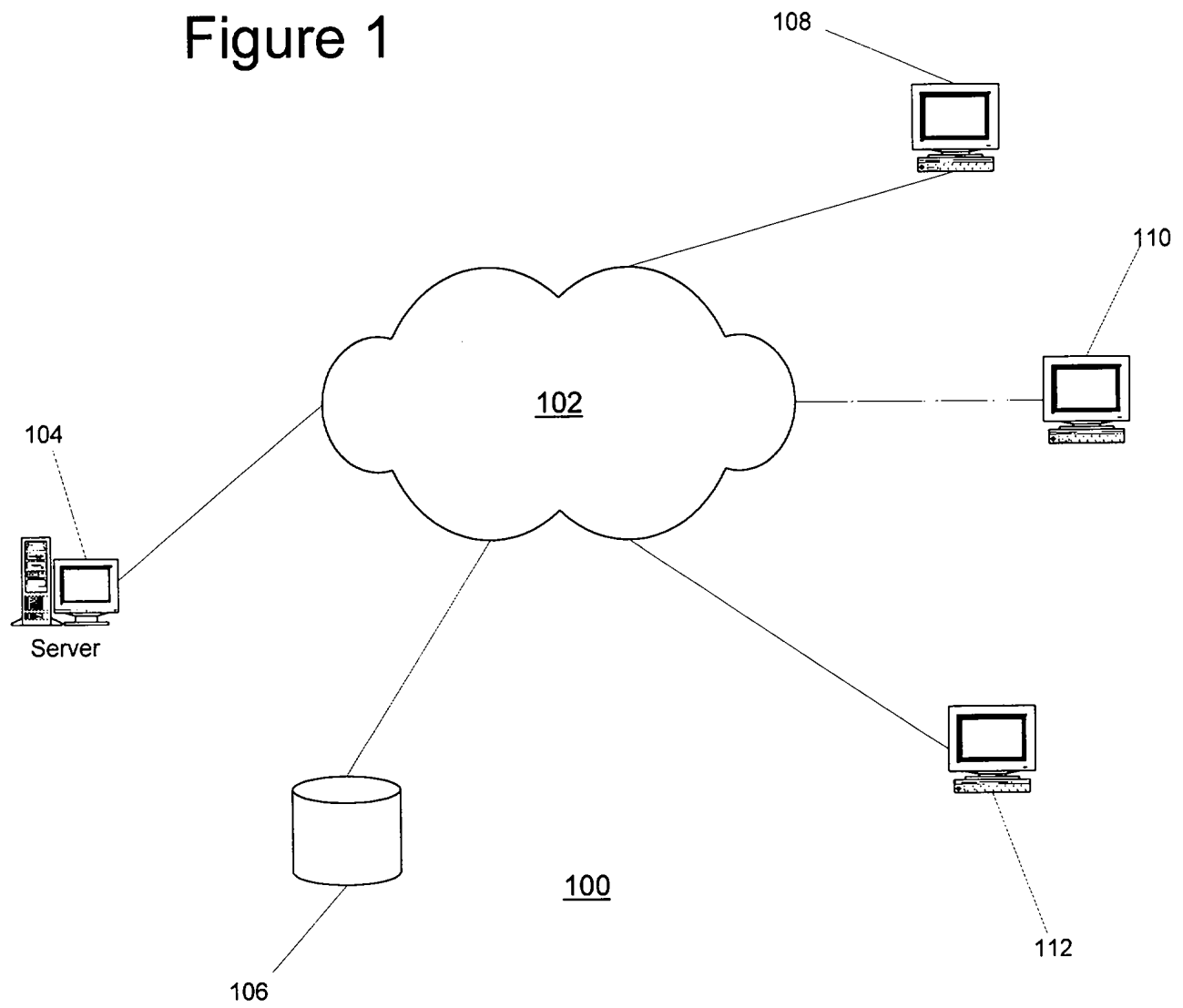
**METHOD AND APPARATUS FOR SELECTIVELY AND DYNAMICALLY  
5           SELECTING CLASSES IN A DATA PROCESSING SYSTEM**

10           A method for selecting classes using a browser for  
            use by a virtual machine in a data processing system.  
            The browser provides an interface in which the interface  
            allows for selection of classes for use by the virtual  
            machine. A selection of classes is received through the  
            interface. The selection of classes is stored by the  
            browser, wherein the selection of classes is used by the  
            browser when initializing the virtual machine.

15

20

Figure 1



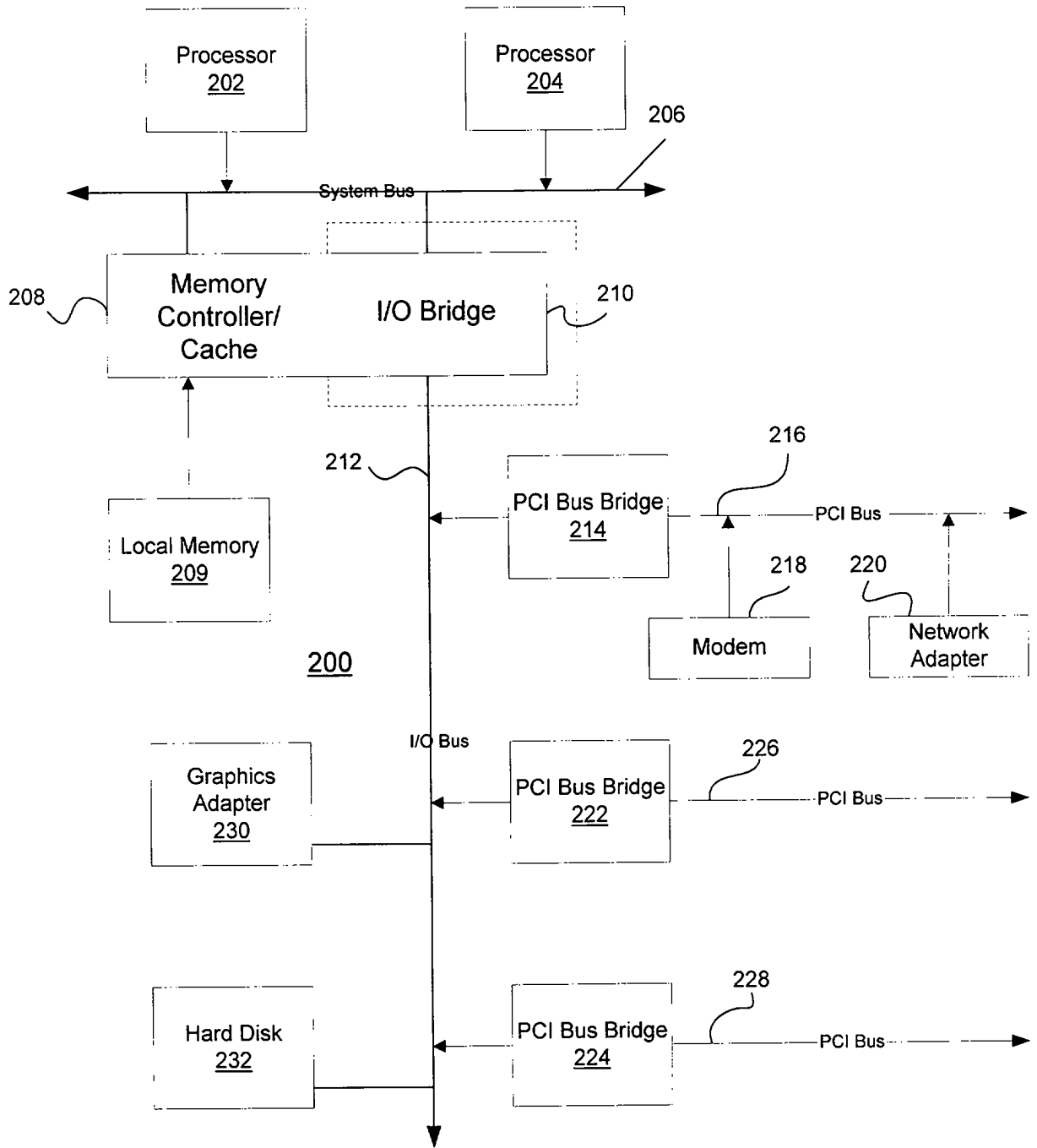


Figure 2

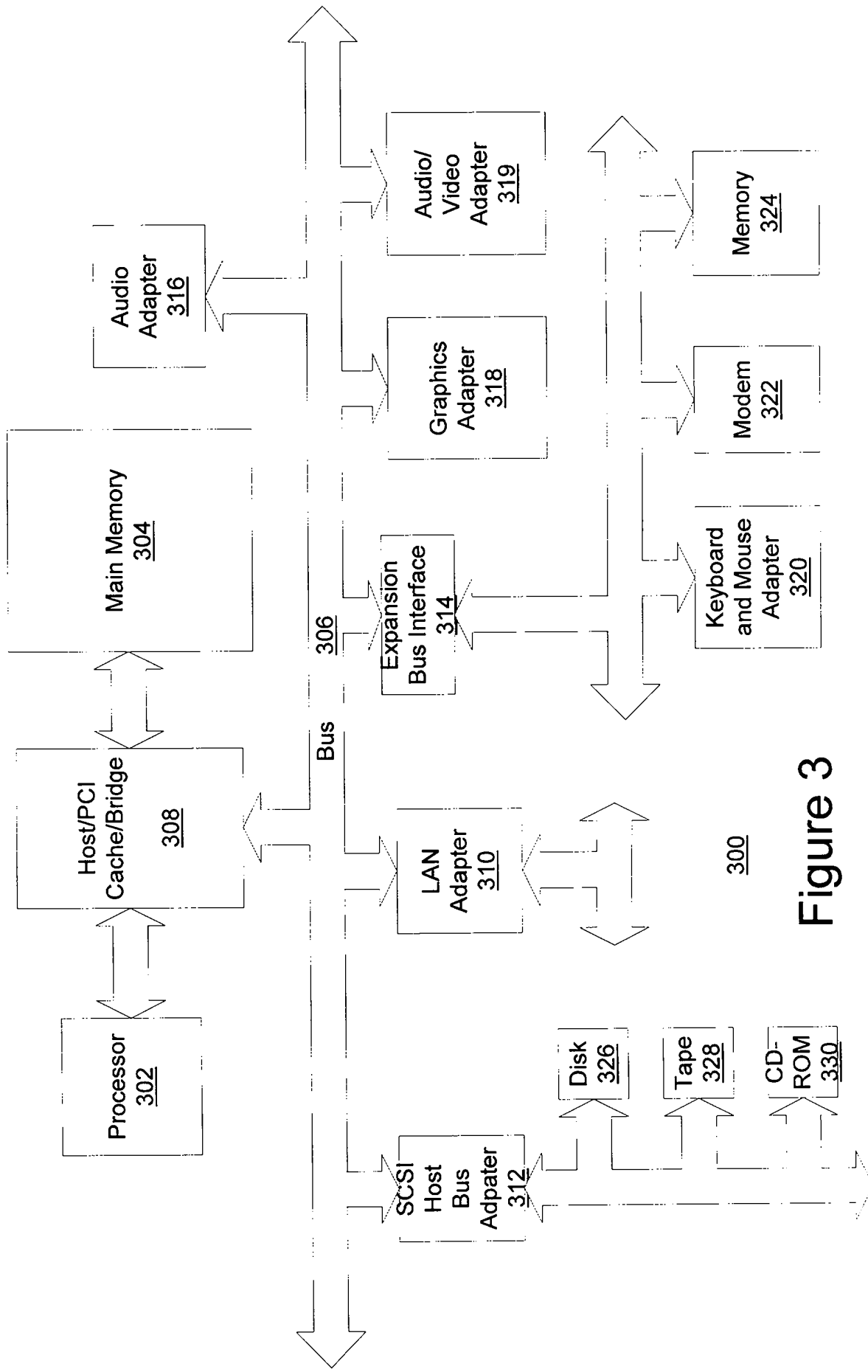


Figure 3

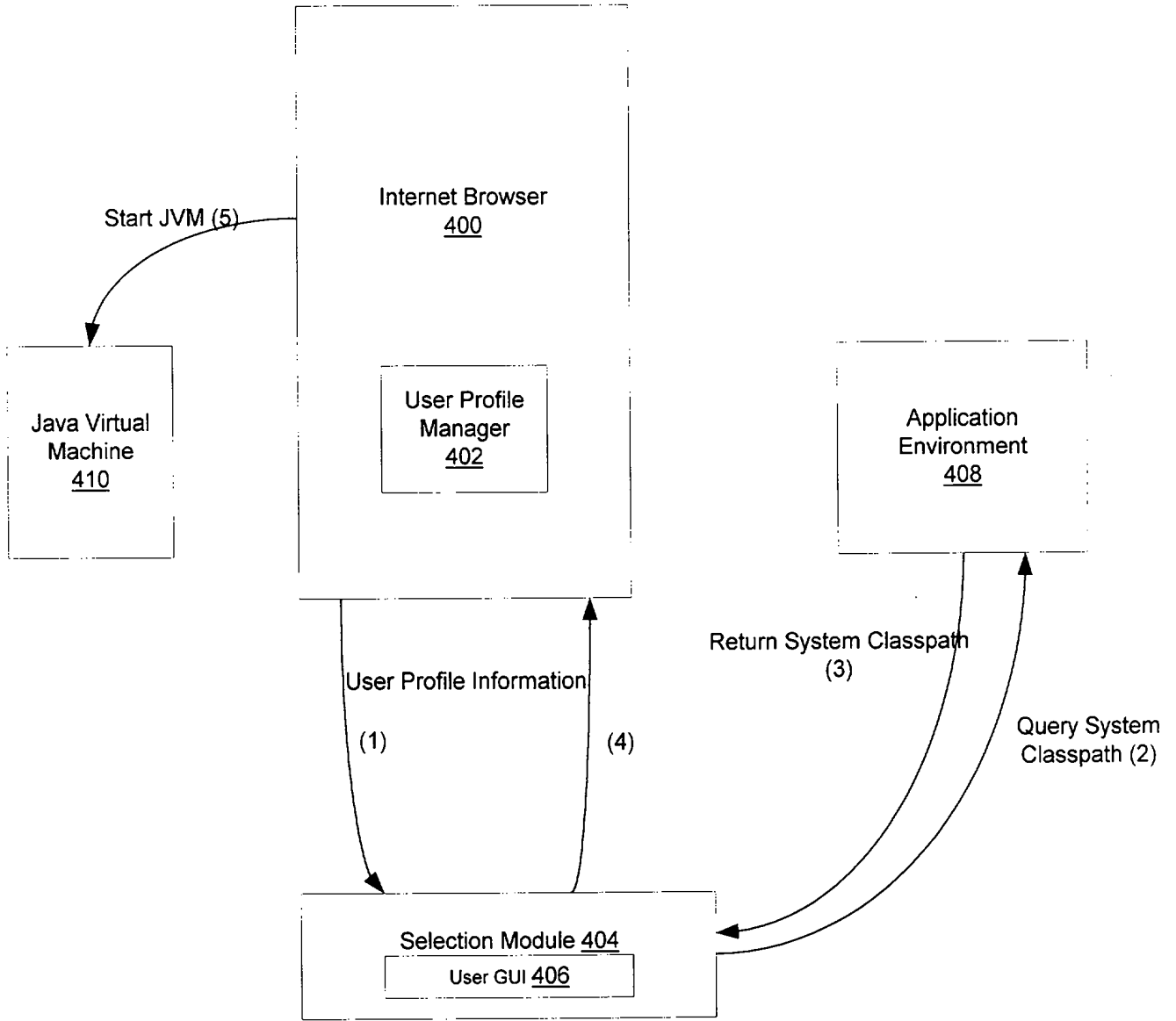


Figure 4

FIGURE 5

IBM Java advanced properties

WARNING: Modifying these properties will change the behavior of JAVA within the browser.

Java Path  
Enter the Path of the Java executable (JVM) to be used.  
C:\Java11\bin\javaw.exe  
Browse

Java Options  
Enter optional Java parameters  
Use of Option

Java Classpath  
☒ Ignore System Classpath  
Enter optional classpath  
c:\swing-1.0.2\swingall.jar  
512

☐ Ignore changes to classpath  
504  
☐ Append to the beginning of the system classpath  
506  
☐ Append to the end of the system classpath

Current Classpath  
The following is the current classpath  
d:\ns\config\os2\bin\nt\_xlib\nt13240.zip;d:\ns\config\os2\bin\nt1\_xlib\nt13240.zip;c:\java11\bin\classes  
508

OK Cancel

512

504

506

508

510

512 915 505

5/9



2025-07-20 14:20:00

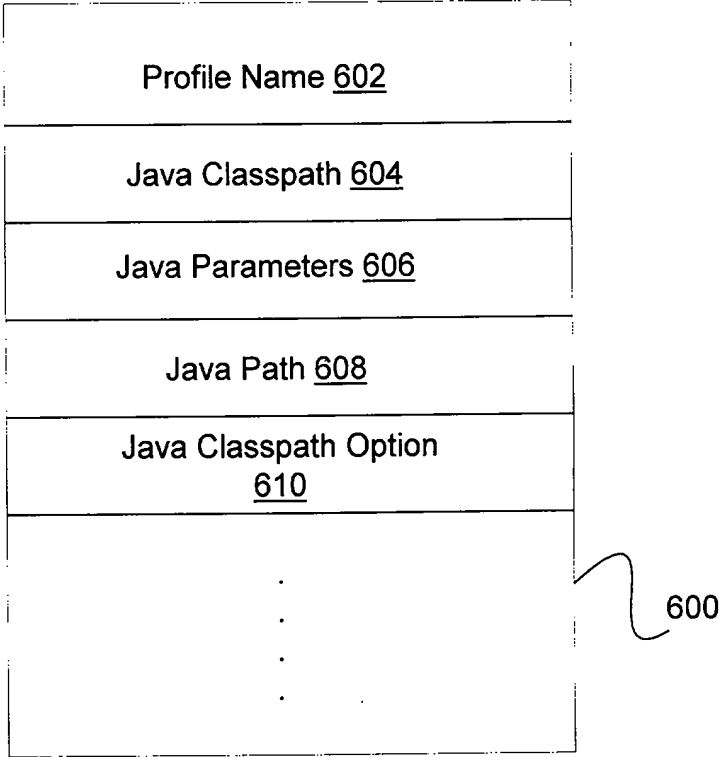


Figure 6

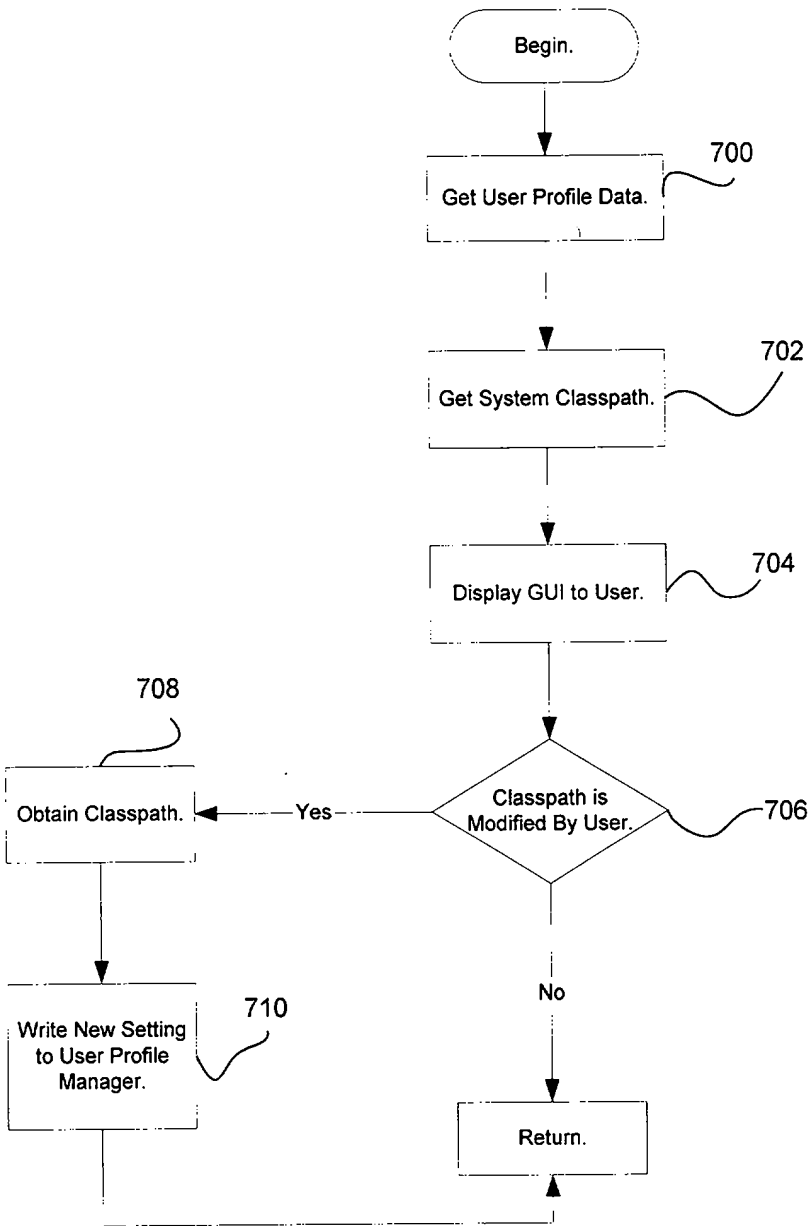


Figure 7

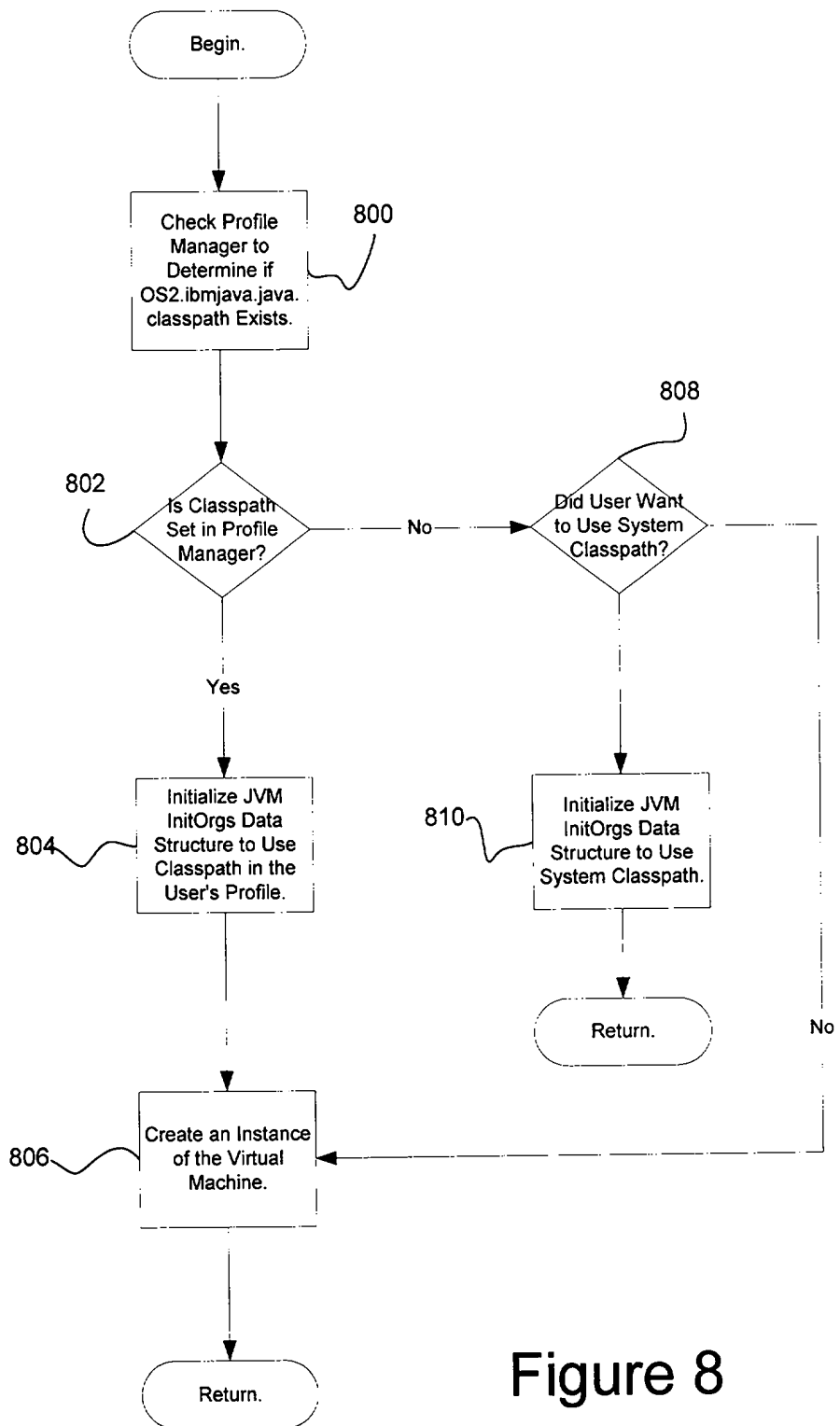


Figure 8

## JVM Initialization Data Structure

*JDK1\_InitArgs Fields*

Field Name	Java Flag	Description
nativeStackSize	-ss	Max Stack Size for native threads
javaStackSize	-oss	Max Stack for any JVM thread
minHeapSize	-ms	Initial heap size
maxHeapSize	-mx	Maximum heap size
verifyMode=0	-noverify	Do not verify byte code when loading
verifyMode=1	-verifyremote	verify byte code only when loading remotely
verifyMode=2	-verify	verify loading of all byte code
classpath	-classpath	Local directories for loading classes
enableClassGC	-nocsge	Enable/disable class garbage collection
enableVerboseGC	-verbosegc	Enable reporting of garbage collection
disableAsyncGC	-noasyncgc	Disable asynchronous garbage collection
verbose	-verbose of -v	Reports JVM information, including all class loads
debugging	-debug	Allow remote debugging of JVM

Figure 9

DECLARATION AND POWER OF ATTORNEY FOR  
PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

METHOD AND APPARATUS FOR SELECTIVELY AND DYNAMICALLY SELECTING CLASSES  
IN A DATA PROCESSING SYSTEM

The specification of which (check one)

X is attached hereto.

— was filed on \_\_\_\_\_  
as Application Serial No. \_\_\_\_\_  
and was amended on \_\_\_\_\_  
(if applicable)

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with Title 37, Code of Federal Regulations, §1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s):  
Claimed

Priority  
Claimed

35729-334450

Yes      No  
(Number)

(Country)

(Day/Month/Year)

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §112, I acknowledge the duty to disclose information material to the patentability of this application as defined in Title 37, Code of Federal Regulations, §1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

(Application Serial #)

(Filing Date)

(Status)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

POWER OF ATTORNEY: As a named inventor, I hereby appoint the following attorneys and/or agents to prosecute this application and transact all business in the Patent and Trademark Office connected therewith.

John W. Henderson, Jr., Reg. No. 26,907; Thomas E. Tyson, Reg. No. 28,543; Robert M. Carwell, Reg. No. 28,499; Jeffrey S. LaBaw, Reg. No. 31,633; Douglas H. Lefevre, Reg. No. 26,193; Mark S. Walker, Reg. No. 30,699; Casimer K. Salys, Reg. No. 28,900; David A. Mims, Jr., Reg. No. 32,708; Richard A. Henkler, Reg. No. 39,220; Volel Emile, Reg. No. 39,969; James H. Barksdale, Jr. Reg. No. 24,091; Anthony V. England, Reg. No. 35,129; Leslie A. Van Leeuwen, Reg. No. P-42,196; Christopher A. Hughes, Reg. No. 26,914; Edward A. Pennington, Reg. No. 32,588; John E. Hoel, Reg. No. 26,279; and Joseph C. Redmond, Jr., Reg. No. 18,753; Duke W. Yee, Reg. No. 34,285, David W. Carstens, Reg. No. 34, 134; and Colin P. Cahoon, Reg. No. 38,836.

Send correspondence to: Duke W. Yee, P.O. Box 802334, Dallas, Texas 75380 and direct all telephone calls to Duke W. Yee, (972) 997-7290.

357629-982334

DOCKET NUMBER: AT9-98-302

FULL NAME OF SOLE OR FIRST INVENTOR: Bruce Anthony Beadle

INVENTORS SIGNATURE: 

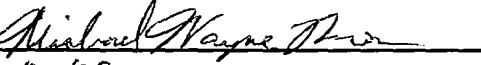
DATE: ~~9-30-98~~ 133 7-30-98

RESIDENCE: 3648 Flora Vista Loop  
Round Rock, Texas 78681

CITIZENSHIP: USA

POST OFFICE ADDRESS: 3648 Flora Vista Loop  
Round Rock, Texas 78681

FULL NAME OF SECOND INVENTOR: Michael Wayne Brown

INVENTORS SIGNATURE: 

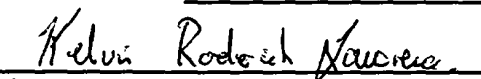
DATE: ~~7/30/98~~ 7/30/98

RESIDENCE: 529 River Down Road  
Georgetown, Texas 78628

CITIZENSHIP: USA

POST OFFICE ADDRESS: 529 River Down Road  
Georgetown, Texas 78628

FULL NAME OF FOURTH INVENTOR: Kelvin Roderick Lawrence

INVENTORS SIGNATURE: 

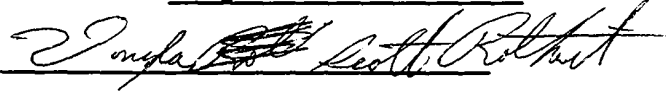
DATE: 7-31-98

RESIDENCE: 1013 Long Cove  
Round Rock, Texas 78664

CITIZENSHIP: United Kingdom

POST OFFICE ADDRESS: 1013 Long Cove  
Round Rock, Texas 78664

FULL NAME OF FIFTH INVENTOR: Douglas Scott Rothert

INVENTORS SIGNATURE: 

DATE: 8/30/98

*DSK*

DOCKET NUMBER: AT9-98-302

RESIDENCE: 11901 Hobby Horse Court, Apt. #1815  
Austin, Texas 78758

CITIZENSHIP: USA

POST OFFICE ADDRESS: 11901 Hobby Horse Court, Apt. #1815  
Round Rock, Texas 78758

FULL NAME OF SIXTH INVENTOR: Robert Michael Russin

INVENTORS SIGNATURE: *Robert Michael Russin*  
DATE: 7/30/98

RESIDENCE: 10701 McFarlie Cove  
Austin, Texas 78750

CITIZENSHIP: USA

POST OFFICE ADDRESS: 10701 McFarlie Cove  
Austin, Texas 78750

06/29/98 09:23:50